PCI Express® Architecture Link Layer Test Specification Revision 2.0

September 17, 2012

Revision	Revision History	DATE
2.0	Initial release.	08/11/2008
	Added acknowledgements section.	09/17/2012

PCI-SIG® disclaims all warranties and liability for the use of this document and the information contained herein and assumes no responsibility for any errors that may appear in this document, nor does PCI-SIG make a commitment to update the information contained herein.

Contact the PCI-SIG office to obtain the latest revision of the specification.

Questions regarding this specification or membership in PCI-SIG may be forwarded to:

Membership Services
www.pcisig.com
E-mail: administration@pcisig.com
Phone: 503-619-0569

Fax: 503-644-6708

Technical Support
techsupp@pcisig.com

DISCLAIMER

This specification is provided "as is" with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. PCI-SIG disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Contents

1

1. Introduction

This test specification primarily covers testing of all PCI Express Port types for compliance with the link layer requirements in Chapter 3 of the PCI Express Base Specification. At this point, this specification does not describe the full set of PCI Express tests for all link layer requirements. Going forward, as the testing gets mature, it is expected that more tests may be added as deemed necessary.

This specification provides a list of test assertions and test definitions pertaining to the link layer. The assertions are simple statements that should be unambiguously answered (binary yes/no, pass/fail) and provide a criteria that these devices must meet for PCI Express compliance testing. Note however that not all assertions may be testable. Test descriptions provide more detailed information (algorithm, test set up, results interpretation, etc.) on how those assertions are tested.

In addition to the test assertions and test definitions specified in this document, devices must also meet other applicable requirements described in the latest versions of the following documents as well as any other criteria required by the PCI-SIG:

PCI Express Architecture PHY Test Specification

PCI Express Architecture Configuration Space Test Specification

Platform BIOS Test Considerations for PCI Express Architecture

PCI Express Architecture Transaction Layer Test Specification

1.1. Abbreviations

BAR – Base Address Register

DUT – Device Under Test

EP – End Point

KEP – Known End Point

PTC – Protocol Test Card

RC – Root Complex

2

2. Test Assertions

Note 1: Test Assertions with Test Descriptions labeled as N/A or TBD are currently not planned for testing in the PCI Express test suite. The assertions must still be met.

Note 2: Test numbers are relevant to this document only and used for reference to test descriptions in this document.. Test numbers do not necessarily relate to a test application executable.

Note 3: Assertion numbers correspond to Checklist numbers with point additions where more than one assertion is needed per Checklist item. Example: Checklist item DLL 3.3#1 would have an Assertion number of DLL 3.3#1 if it is the only assertion for that checklist item. In the event more than one assertion is necessary, the assertion numbers would be DLL 3.3#1 and DLL 3.3#1.1, DLL3.3#1.2, and so on.

Note 4: There are published PCI Express 1.1 checklists. The 1.1 assertions listed here are the assertions from the checklists documents tested by configuration test descriptions. The checklists were not updated for PCI Express 2.0.

2.1. Table of Assertions

Assertion #	Assertion Description	Test #
Link Layer Overview		
Data Link and Control and Management State Machine		
DLL.4.1#2	Upon Receiving a DLLP, the data link layer receiver state machine must ignore reserved fields.	Test41-20
Flow Control Initialization Protocol		
Data Link Layer Packet Rules		
LCRC and Sequence Number Rules (TLP Transmitter)		
DLL.5.2#1	The link transmitter must start REPLAY of its RETRY_BUF as soon as (after finishing the current TLP in progress) a NAK is received.	Test 52-10

Assertion #	Assertion Description	Test #
DLL.5.2#1.1	The link transmitter must start REPLAY of its RETRY_BUF upon its REPLAY_TIMER expiring when there are still some TLPs which have not received Ack or Nak DLLPs.	Test52-11
DLL.5.2#1.2	A TLP must be retransmitted until a positive acknowledgement has been sent by the receiver and received by the transmitter and REPLAY_NUM does not overflow while retransmitting (subject to the timeout values in Table 3.4 and Table 3.5 to which Rx/Tx_L0s_Adjustment must be applied as appropriate).	Test52-12
DLL.5.2#2	If repeated retries fail and REPLAY_NUM overflows, the link transmitter must ask the Physical Layer to retrain the link. There must be a reported error to correspond to this as per Section 6.2.	Test52-20
DLL.5.2#7	The link state (including the contents of retry buffer) must not change in link retraining if Physical LinkUp=1.	Test52-20
DLL.5.2#10	Oldest unacknowledged TLP must be sent first in replay and followed by the rest of the unacknowledged TLPs in the original transmission order.	Test52-100
DLL.5.2#15	All corrupt DLLPs must be discarded and be reported as an error associated with the port.	Test52-150
DLL.5.2#16	A DLLP with undefined encodings shall be dropped silently by the receiver with no error associated.	Test52-160
DLL.5.2#17	If an Ack DLLP does not have a sequence number of an unacknowledged TLP, or of the most recently acknowledged TLP, it must be reported as a DL Layer protocol error associated with the port.	Test52-170
LCRC and Sequence Number (TLP Receiver)		
DLL.5.3#2	If a normal TLP is received and its LCRC does not match calculated CRC, discard the TLP, free any storage associated with it, schedule a NAK DLLP for transmission if one is not already scheduled and report an error associated with the Port	Test53-20
DLL.5.3#3.1	If a TLP is received with EDB end framing symbol, and the LCRC is *not* the logical NOT of the calculated value, discard the TLP, free any storage associated with it, schedule a NAK DLLP for transmission if one is not already scheduled and report an error associated with the Port.	Test 53-31

3

3. General Testing Overview and Topologies

Advanced Error reporting is an optional capability as per the PCI Express Base Specification. However, if a DUT implements such capability, it must meet the requirements laid out in the PCI Express Base Specification. Hence in the test definitions, the check for advanced error reporting shall be applicable only if the DUT implements it.

In order to test at the link layer, it is assumed that the test equipment will have the following capabilities

1) The test equipment allows any DUT up to x16 PCI Express lane width to be connected to the golden system
2) The DUT will operate normally (though perhaps at reduced performance levels) with its drivers loaded and all applicable applications executable on it with the right configuration.
3) The test equipment has at least the following capabilities and more as testing requires:
    a. Intentionally NAK matching TLPs from the DUT
    b. Intentionally ignore (no ACK or NAK) matching TLPs from the DUT
    c. Delay ACK or NAK within legal and illegal boundaries
    d. Drop and Delay matching TLPs going to and from the DUT
    e. Corrupt different CRCs and Digest fields of TLPs and DLLPs to and from the DUT
    f. Generate bogus sequence numbers in ACKs and NAKs in DLLPs to and from the DUT
    g. Generate and Receive traffic from the platform – have basic Master/Target memory and message capabilities
    h. Give test software control on when the test conditions are to be applied through the ARM and DISARM. The test equipment will only apply the test conditions between ARM and DISARM.
    i. Test equipment has some memory resources (at least 4 KB) behind one of its BARs (Base Address Register) and the test software can use it as either scratch pad memory or trace buffer memory.
    j. Get the status from the test equipment on where it is in the process of applying test conditions. In this document, it is assumed that the test equipment can assert the test conditions for multiple times (programmable) and the status indicates how many times the test equipment still needs to assert them. A count of zero indicates the test equipment has asserted the test conditions.

k. The test equipment can apply similar test conditions to a platform under test or an add-in card under test. Hence it shall have suitable operating modes.

Going forward, test equipment with the above characteristics and capabilities will be referred to as the Protocol Test (PTC) card in this document. In order to facilitate testing of both platforms and add-in cards, the PTC will have the following modes of operation:

1) Platform Test Mode – PTC will apply the test conditions to a root port (DUT) that it is connected to. The over all topology (platform + PTC) is referred to as the Platform Test Topology and is as shown below.

2) Add-Card Test Mode – PTC will apply the test conditions to a device on an add-in card plugged into the PTC. The add-in card will include an Endpoint (EP) or a PCI Express Switch or a Bridge that constitutes the DUT. The topology (platform +PTC + add-in card) is referred to as Endpoint Test Topology.

   a. While testing a PCI Express Switch upstream the same Endpoint Test Topology applies. However when an upstream port of a switch is being tested, the order of the PTC and the switch add-in card are interchanged and that topology is called Switch Test Topology.

The relevant Macros for use with PTC are defined in Appendix A. The purpose of the Macros is to make the test definition writing consistent and focus on the functionality rather than on any test equipment implementation. Examples of test topology are shown in the figures that follow.

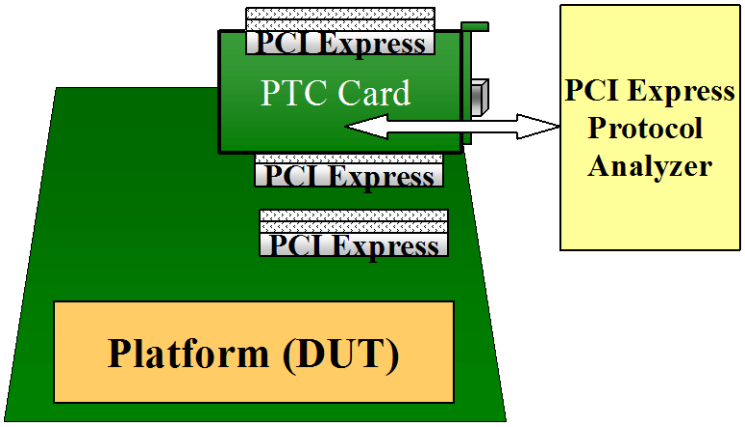

Figure 1: Platform Test Topology

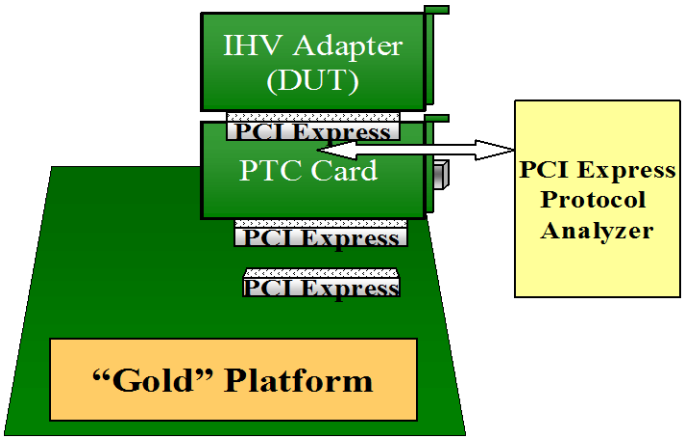

Figure 2: Endpoint Test Topology

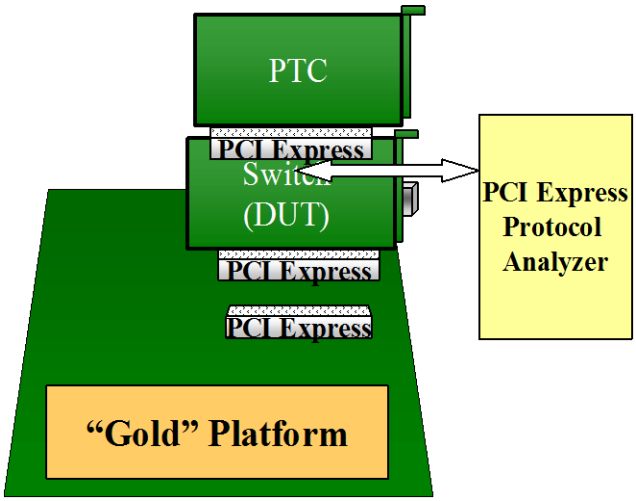

Figure 3: Switch Test Topology

4

4. Test Descriptions

All tests are run at both 2.5 GT/s and 5.0 GT/s for add in cards that support 5.0 GT/s.

When testing multi-function endpoint DUTs, error reporting shall only be enabled (when required) in the Device Control register for the function currently being tested, and disabled for all the other functions on the DUT.

4.1. Link Layer Overview

4.2. Data Link Layer Packet Rules

4.2.1. Test 41-20 ReservedFieldsDLLPReceive ↑

Test Introduction

The intent of this test is to verify that the DUT truly ignores reserved fields in an ACK DLLP by sending arbitrary data in those fields.

ASSERTIONS COVERED: DLL.4.1#2

NOTES:

1) Test applies to all PCI Express port types.
2) Test could optionally be extended to cover all DLLP types as deemed necessary.
3) DATA_BUF – holds the data read back from the device

4.2.1.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and PTC is the completer for that request in this test

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (CORRUPT_RESERVED_FIELDS_ACK_DLLP, CONFIG_RD_REQ, 1) // PTC will use non-zero values in at least one reserved field of ACK for this TLP.
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO_PTC_CLEANUP ()
6) Verify that the DUT ignores the non-zero reserved values in the ACK and did not retransmit the same CONFIG_RD_REQ TLP. If the DUT meets these criteria, the DUT passes the test.
7) If the DUT did retransmit the TLP as above, log it as DUT's failure.

4.2.1.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (CORRUPT_RESERVED_FIELDS_ACK_DLLP, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. DUT ignored the non-zero field(s) in the ACK and did not retransmit the CONFIG_RD_COMPLETION TLP.
      ii. Verify that the DUT did not set any error bits in Device Status register.

i. If the DUT meets above criteria, the DUT passes the test for Function 0.

j. If the DUT did not ignore the reserved fields, and retransmitted the TLP as above, or if any error detected bits are set, log it as DUT's failure.

2) For (Function =1; Function=7; Function++) in DUT:

a. Clear Device Status register and verify that none of the error status bits are set.

b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

c. MACRO_PTC_PROGRAM (CORRUPT_RESERVED_FIELDS_ACK_DLLP, ANY_TLP, 1) // ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

i. DUT ignored the non-zero field(s) in the ACK and did not retransmit the CONFIG_RD_COMPLETION TLP or Unsupported Request TLP depending on whether the function exists or not.

ii. Verify that the DUT did not set any error bits in device status register other than Unsupported Request Detected (bit 3).

i. If the DUT meets above criteria, the DUT passes the test.

j. If the DUT did not ignore the reserved fields, and retransmitted the TLP as above, or if any error detected bits are set, log it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.2.1.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.2.1.4. Switch Upstream Port Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3. LCRC and Sequence Number Rules (TLP Transmitter)

4.3.1. Test 52-10 RetransmitOnNak ↑

Test Introduction

The intent of this test is to ensure that a DUT will retransmit a transaction for which a NAK has been issued.

ASSERTIONS COVERED: DLL.5.2#1

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.1.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and PTC is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (NAK, CONFIG_RD_REQ, 1)
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO_PTC_CLEANUP ()
6) Verify that the DUT retransmits the CONFIG_RD_REQ TLP for which it received the NAK. If it did, the DUT passes the test.
7) If the DUT did not retransmit the TLP as above, log it as DUT's failure.

4.3.1.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (NAK, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. CONFIG_RD_COMPLETION TLP for which a NAK is received, is retransmitted by the DUT.
      ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in device status register.
      iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.
      iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.
   i. If all of the conditions in the above step are met, DUT PASSES the test for Function 0.
2) For (Function=1;Function =7;Function++) in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (NAK, ANY_TLP, 1) // ANY_ TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

   i. CONFIG_RD_COMPLETION TLP or Unsupported Request TLP (depending on whether the function exists or not) for which a NAK is received, is retransmitted by the DUT.

   ii. Only Correctable Error bit (bit 0) or Unsupported request bit(bit 3) in the DUT's Device Status register (depending on whether the Function exists or not) is set and DUT did not set any other error bits in device status register.

   iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.

   iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

i. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.1.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.1.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.2. Test 52-11 REPLAY_TIMERTest ↑

Test Introduction

The intent of this test is to ensure that a DUT's REPLAY_TIMER is working properly by not sending neither an ACK nor a NAK.

ASSERTIONS COVERED: DLL.5.2#1.1

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.2.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and the PTC is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (NO_ACK_NAK, CONFIG_RD_REQ, 1)
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO_PTC_CLEANUP ()
6) Verify that the DUT retransmits the CONFIG_RD_REQ TLP for which it received no ACK or NAK. If it did, the DUT passes the test.
7) If the DUT did not retransmit the TLP as above, log it as DUT's failure

4.3.2.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request in this test

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (NO_ACK_NAK, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.
      ii. CONFIG_RD_COMPLETION TLP is retransmitted by the DUT.
      iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.
      iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.
   i. If all of the conditions in the above step are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.
2) For (Function=1; Function=7; Function++) in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (NO_ACK_NAK, ANY_TLP, 1)//ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

   i. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3 – depending on whether the function exists or not) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

   ii. CONFIG_RD_COMPLETION TLP or Unsupported Request TLP (depending on whether function exists or not) is retransmitted by the DUT.

   iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

   iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

i. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.2.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.2.4. Switch Upstream / Bridge Port Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.3. Test 52-12 REPLAY_NUMTest ↑

Test Introduction

The intent of this test is to ensure that a DUT will keep retransmitting a transaction for which a NAK has been issued on purpose until the number of times its REPLAY_NUM supports.

ASSERTIONS COVERED: DLL.5.2#1.1

NOTES:

1) Test applies to all PCI Express port types.
2) REPLAY_NUM – holds the maximum number of times a DUT can retransmit a TLP without causing link retraining
3) DATA_PATTERN – 0xa5b4c3d2 (arbitrarily chosen)
4) BYTE_COUNT – number of bytes written into the PTC memory with the specified pattern – use value of one (forces a single TLP in all cases that can be NAKed)
5) DATA_BUF – holds the data read back from the device

4.3.3.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and the PTC is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) REPLAY_NUM = Determine (ask user if needed) how many times the DUT can retransmit a TLP before requesting link retraining. If not obtainable, use 3 for this count.
2) MACRO_PTC_PROGRAM (NAK, CONFIG_RD_REQ, REPLAY_NUM)
3) MACRO_PTC_ARM()
4) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
5) MACRO_PTC_STATUS (ACTION COUNT)
6) MACRO_PTC_CLEANUP ()
7) Verify that the DUT transmits the CONFIG_RD_REQ TLP for REPLAY_NUM of times. If it did, the DUT passes the test.
8) If the DUT did not retransmit the TLP as above, log it as DUT's failure.

4.3.3.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode.

SECTION NOTES:

1) RC is the CONFIG_RD requester and the EP (DUT) is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. REPLAY_NUM = Determine (ask user if needed) how many times the DUT can retransmit a TLP before requesting link retraining. If not obtainable, use 3 for this count.
   c. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   d. MACRO_PTC_PROGRAM (NAK, CONFIG_RD_COMPLETION, REPLAY_NUM)
   e. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   f. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   g. MACRO-PTC_DISARM ()
   h. MACRO_READ_DATA_FROM_PTC( )
   i. Verify that the DUT transmits the CONFIG_RD_COMPLETION for REPLAY_NUM of times with the same sequence number and no error detected bits are set in the device.
   j. If the DUT meets above criteria, the DUT passes the test for Function 0.

2) For (Function=1; Function=7; Function++) in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. REPLAY_NUM = Determine (ask user if needed) how many times the DUT can retransmit a TLP before requesting link retraining. If not obtainable, use 3 for this count.
   c. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   d. MACRO_PTC_PROGRAM (NAK, ANY_TLP, REPLAY_NUM)//ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.
   e. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   f. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

g. MACRO-PTC_DISARM ()

h. MACRO_READ_DATA_FROM_PTC( )

i. Verify that the DUT transmits the CONFIG_RD_COMPLETION or Unsupported Request TLP (depending on whether the Function exists or not) for REPLAY_NUM of times with the same sequence number and no error bits except for Unsupported Request bit (bit 3 depending on whether function exists or not) is set in the device.

j. If the DUT meets above criteria, the DUT passes the test.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.3.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.3.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.4. Test 52-20 LinkRetrainOnRetryFail ↑

Test Introduction

The intent of this test is to ensure that the link connected to the DUT will go into retraining after trying for REPLAY_NUM of times to get a TLP through and failing. It will also test that while in retraining the retry buffer and link states are not changed and the pending TLP is retransmitted after link retraining completes.

ASSERTIONS COVERED: DLL.5.2#2, DLL.5.2#7

NOTES:

1) Test applies to all PCI Express port types.

2) REPLAY_NUM – holds the maximum number of times a DUT can retransmit a TLP without causing link retraining. Use one more than this to cause retraining of link while still keeping PhysicalLinkUp = 1.

3) DATA_BUF – holds the data read back from the device

4.3.4.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the MEM_RD requester and PTC is the completer for that request.
2) This memory read request target is the memory behind the PTC allocated through the BAR mechanism.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) REPLAY_NUM = Determine (ask user if needed) how many times the RC can retransmit a TLP before requesting link retraining. If not obtainable, use 5 for this count.
2) MACRO_PTC_PROGRAM (NAK, CONFIG_RD_REQ, REPLAY_NUM+1)
3) MACRO_PTC_ARM ()
4) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
5) MACRO_POLL_DUT_FOR_LINK_RETRAINING () // of the root port link status register connected to the PTC.
6) Verify that link got successfully retrained through Link Status register of the DUT (confirm via a protocol analyzer if needed) and an error is logged in the DUT's port register for REPLAY_NUM overflow (optionally verify that an error message is generated by the DUT if enabled). If not, treat it as DUT failure - DLL.5.2#2.
7) MACRO_PTC_STATUS (POLL) // verify the action count is zero
8) MACRO-PTC_CLEANUP ()
9) Verify that the link did not change its state. That is the link should be in active state while undergoing link retraining (DLL.5.2#7) (by checking to see that that the TS1 and TS2 sequences keep the lane and link numbers intact while retraining, and flow credit DLLPs are exchanged instead of InitFC DLLPs). If not, treat it as DUT's failure- DLL.5.2#7.
10) Verify that DATA_BUF contains the Vendor and Device Ids of the PTC, written after link got retrained. If not, treat it as DUT's failure-DLL.5.2#7.

4.3.4.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and the EP (DUT) is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. REPLAY_NUM = Determine (ask user if needed) how many times the DUT can retransmit a TLP before requesting link retraining. If not obtainable, use 3 for this count.
   c. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   d. MACRO_PTC_PROGRAM (NAK, CONFIG_RD_COMPLETION, REPLAY_NUM+1)
   e. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   f. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   g. MACRO_POLL_PTC_FOR_LINK_RETRAINING()//of the port connected to the DUT
   h. MACRO-PTC_DISARM ()
   i. MACRO_READ_DATA_FROM_PTC( )
   j. Verify that:

      CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

      i. Link got successfully retrained through Link Status register of the PTC (confirm via a protocol analyzer if needed), an error is logged in the DUT's port register for REPLAY_NUM overflow and that an error message is generated by the DUT. If not, treat it as DUT failure - DLL.5.2#2.

      ii. Verify that DUT retransmitted Config data after the link retraining. If not, treat it as DUT failure - DLL.5.2#7.

      iii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

      iv. If the error is *not* masked in the Correctable Error Mask register (bit 8) of AER:

         a) REPLAY_NUM Rollover status bit (bit 8) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If the error is masked in the Correctable Error Mask register (bit 8) of AER, then *no* ERR_COR is sent.

vi. If all of the conditions in the above step are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 -Correctable Error – No AER implemented

i. Link got successfully retrained through Link Status register of the PTC (confirm via a protocol analyzer if needed), an error is logged in the DUT's port register for REPLAY_NUM overflow and that an error message is generated by the DUT. If not, treat it as DUT failure - DLL.5.2#2.

ii. Verify that DUT retransmitted Config data after the link retraining. If not, treat it as DUT failure - DLL.5.2#7.

iii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iv. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

v. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

vi. If all of the conditions in the above step are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

2) For (Function=1;Function=7;Function++) in DUT:

a. Clear Device Status register and verify that none of the error status bits are set.

b. REPLAY_NUM = Determine (ask user if needed) how many times the DUT can retransmit a TLP before requesting link retraining. If not obtainable, use 3 for this count.

c. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

d. MACRO_PTC_PROGRAM (NAK, ANY_TLP, REPLAY_NUM+1) //ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

e. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

f. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

g. MACRO_POLL_PTC_FOR_LINK_RETRAINING()//of the port connected to the DUT

h. MACRO-PTC_DISARM ()

i. MACRO_READ_DATA_FROM_PTC( )

j. Verify that:

CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

i. Link got successfully retrained through Link Status register of the PTC (confirm via a protocol analyzer if needed), an error is logged in the DUT's port register for REPLAY_NUM overflow and that an error message is generated by the DUT. If not, treat it as DUT failure - DLL.5.2#2.

ii. Verify that DUT retransmitted Config data or Unsupported Request(depending on whether the function exists or not) after the link retraining. If not, treat it as DUT failure - DLL.5.2#7.

iii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3 - depending on whether the function exists or not) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iv. If the error is *not* masked in the Correctable Error Mask register (bit 8) of AER:

a) REPLAY_NUM Rollover status bit (bit 8) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If the error is masked in the Correctable Error Mask register (bit 8) of AER, then *no* ERR_COR is sent.

vi. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 -Correctable Error – No AER implemented

i. Link got successfully retrained through link status register of the PTC (confirm via a protocol analyzer if needed), an error is logged in the DUT's port register for REPLAY_NUM overflow and that an error message is generated by the DUT. If not, treat it as DUT failure - DLL.5.2#2.

ii. Verify that retransmitted Config data or Unsupported Request(depending on whether the function exists or not) after the link retraining. If not, treat it as DUT failure - DLL.5.2#7.

iii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3 - depending on whether the function exists or not) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iv. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

v. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

vi. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.4.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.4.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.5. Test 52-100 ReplayTLPOrder ↑

Test Introduction

The intent of this test is to verify that the oldest unacknowledged TLP is retransmitted first in replay followed by the other unacknowledged TLPs in the same order they were transmitted first.

ASSERTIONS COVERED: DLL.5.2#10

NOTES:

1) Test applies toall PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.5.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) This test assumes that since no ACK or NAK has been issued to any of the TLPs, all of them will be retried.
2) RC (DUT) is the requester and the PTC is the completer for any request in this test.
3) This memory read request target is the memory behind the PTC allocated through BAR mechanisms.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (NO_ACK_NAK, CONFIG_RD_REQ, REPLAY_NUM)
2) MACRO_PTC_ARM ()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (PCI_COMPATIBLE_SPACE) // first 24 bytes in DWORD fashion
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO-PTC_CLEANUP ()
6) Verify that the trace contains at least one or more replays of multiple TLPs, sent in the original order. If the replays satisfy the order condition, the DUT passes the test.
7) If the replay order does not match, treat it as DUT's failure.

4.3.5.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and the EP (DUT) is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.
4) Link training identified in its InitFCs that the endpoint supports two or more NP credits. If that is not the case, the test passes with according warning that the test couldn't be executed.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (NO_ACK_NAK, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

i. DUT retransmitted CONFIG_RD_COMPLETION TLPs.

ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

i. If all of the conditions in the above step are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

2) For (Function=1;Function=7;Function++) in DUT:

a. Clear Device Status register and verify that none of the error status bits are set.

b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

c. MACRO_PTC_PROGRAM (NO_ACK_NAK, ANY_TLP, 1) // ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

i. DUT retransmitted CONFIG_RD_COMPLETION TLPs or Unsupported Request TLP depending on whether functions exist or not.

ii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

i. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.5.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.5.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.6. Test 52-150 CorruptedCRC_DLLP ↑

Test Introduction

The intent of this test is to ensure that a DUT recognizes a DLLP with bad CRC, drops it and logs a BAD_DLLP port error.

ASSERTIONS COVERED: DLL.5.2#15

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.6.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and the PTC is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (CORRUPT_ACK_CRC, CONFIG_RD_REQ, 1)
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)

5) MACRO_PTC_CLEANUP ()
6) Verify that the DUT retransmits the CONFIG_RD_REQ TLP for which it has received an ACK with bad CRC and logs a BAD_DLLP error associated with that port. If it did, the DUT passes the test.
7) If the DUT did not retransmit the TLP as above, log it as DUT's failure.

4.3.6.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (CORRUPT_ACK_CRC, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:

   CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

      i. The CONFIG_RD_COMPLETION TLP for which the DUT received an ACK with bad CRC is retransmitted by the DUT.
      ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If the error is *not* masked in the Correctable Error Mask register (bit 7) of AER:

a) Bad DLLP status bit (bit 7) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

iv. If the error is masked in the Correctable Error Mask register (bit 7) of AER, then no ERR_COR is sent.

v. If all the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error – No AER implemented

i. The CONFIG_RD_COMPLETION TLP for which the DUT received an ACK with bad CRC is retransmitted by the DUT.

ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If all the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

2) For (Function=7;Function=7;Function++) in DUT:

a. Clear Device Status register and verify that none of the error status bits are set.

b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

c. MACRO_PTC_PROGRAM (CORRUPT_ACK_CRC, ANY_TLP, 1) // ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

i. The CONFIG_RD_COMPLETION TLP or Unsupported Request TLP (depending on whether function exists or not) for which the DUT received an ACK with bad CRC is retransmitted by the DUT.

ii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If the error is *not* masked in the Correctable Error Mask register (bit 7) of AER:

a) Bad DLLP status bit (bit 7) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

iv. If the error is masked in the Correctable Error Mask register (bit 7) of AER, then *no* ERR_COR is sent.

v. If all the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error – No AER implemented

i. The CONFIG_RD_COMPLETION TLP or Unsupported Request TLP (depending on whether function exists or not) for which the DUT received an ACK with bad CRC is retransmitted by the DUT.

ii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If all the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.6.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.6.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.7. Test 52-160 UndefinedDLLPEncoding ↑

Test Introduction

The intent of this test is to verify that the DUT silently drops any DLLP with undefined encoding (any pattern for DLLP type that is reserved right now) and no error is associated with it.

ASSERTIONS COVERED: DLL.5.2#16

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.7.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and the PTC is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) Read the DUT's advanced error reporting registers and save the values.
2) MACRO_PTC_PROGRAM (DLLP_UNDEFINED_ENCODING, CONFIG_RD_REQ, 1)
3) MACRO_PTC_ARM()
4) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
5) MACRO_PTC_STATUS (ACTION COUNT)
6) MACRO_PTC_CLEANUP ()
7) Verify that the DUT silently drops the DLLP with undefined encoding – read the DUT's advanced error reporting registers and verify that they did not change from earlier read values (except for REPLAY_TIMER overflow being set). If they did not change, the DUT passes the test.

8) Verify that the DUT retries the configuration read after its ACK_NAK latency timer expires and the transaction is completed successfully. If it did, the DUT passes the test.

9) If not , log it as DUT's failure.

4.3.7.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.

2) PTC is disarmed and no trigger conditions set up.

3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT;
   a. Clear Device Status register and verify that none of the error status bits are set
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (DLLP_UNDEFINED_ENCODING, CONFIG_RD_COMPLETION, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. DUT retransmits the CONFIG_RD_COMPLETION TLP for which it received an undefined DLLP.
      ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.
      iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.
      iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.
   i. If all of the conditions in the above step are met, DUT PASSES the test for function 0. Otherwise consider it as DUT's failure.

2) For (Function =1;Function=7;Function++) in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (DLLP_UNDEFINED_ENCODING, ANY_TLP, 1) //ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. DUT retransmits the CONFIG_RD_COMPLETION TLP or Unsupported Request TLP(depending on whether functions exists or not) for which it received an undefined DLLP.
      ii. Only Correctable Error bit (bit 0) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in device status register.
      iii. If error reporting is enabled in DUT's Device Control register (bit 0), send ERR_COR message.
      iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.
   i. If all of the conditions in the above step are met, DUT PASSES the test. Otherwise consider it as DUT's failure.
3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.7.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.7.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.3.8. Test 52-170 WrongSeqNumInAckDLLP ↑

Test Introduction

The intent of this test is to verify that the DUT drops any ACK DLLP that does not have a sequence number corresponding to an unacknowledged TLP and logs a Data Link Protocol error associated with the port.

ASSERTIONS COVERED: DLL.5.2#17

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.3.8.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the CONFIG_RD requester and the PTC is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (ACK_DLLP_WRONG_SEQ_NUM, CONFIG_RD_REQ, 1) // PTC will supply an incorrect sequence number to the configuration read request TLP from RC.
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO_PTC_CLEANUP ()
6) Verify that the DUT logs BAD DLLP error. If it did, the DUT passes the test.
7) If the DUT did not report an error, log it as DUT's failure.

4.3.8.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and EP (DUT) is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_ PROGRAM (ACK_DLLP_WRONG_SEQ_NUM, CONFIG_RD_COMPLETION, 1) // PTC will issue an ACK with incorrect sequence number in response to the completion TLP.
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:

   CASE 1: PCIe1.0.a - Fatal Error - Severity – Non-Fatal - AER implemented

   This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is cleared to indicate that this is *not* a fatal error.

      i. Only Non-Fatal Error bit (bit 1) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.
      ii. If the error is *not* masked in the Uncorrectable Error Mask register (bit 4) of AER:
         a) Data Link Protocol error bit (bit 4) in the Uncorrectable Error Status register of the AER is set.
         b) If error reporting is enabled in DUT's Device Control register, send ERR_NONFATAL message.
         c) If error reporting is not enabled in DUT's Device Control register, *no* ERR_NONFATAL is sent.
      iii. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_NONFATAL is sent.

iv. If all the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a - Fatal Error - Severity – Fatal - AER implemented

This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is set to indicate that this is a fatal error.

i. Only Fatal Error bit (bit 2) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

ii. If the error is *not* masked in the Uncorrectable Error Mask register (bit 4) of AER:

   a) Data Link Protocol error bit (bit 4) in the Uncorrectable Error Status register of the AER is set.

   b) If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

   c) If error reporting is not enabled in DUT's Device Control register, *no* ERR_FATAL is sent.

iii. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_FATAL is sent.

iv. If all of the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 3: PCIe1.0.a - Fatal Error - No AER

This implies DUT has no AER and the Malformed TLP is handled as a fatal error.

i. Only Fatal Error bit (bit 2) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

ii. If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

iii. If error reporting is not enabled in DUT's Device Control register, *no* ERR_FATAL is sent.

iv. If all of the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 4: PCIe1.1 - Fatal Error - No AER

PCIe2.0 - Fatal Error - No AER

This implies DUT has no AER and the Malformed TLP is handled as a fatal error.

i. Only Fatal Error bit (bit 2) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

ii. If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

iii. If error reporting is not enabled in DUT's Device Control register, no ERR_FATAL is sent.

iv. If all of the above conditions are met, DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 5: PCIe1.1 – Fatal Error – Severity – Non-Advisory - AER implemented

PCIe2.0 – Fatal Error – Severity – Non-Advisory - AER implemented

This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is set to indicate that this is a fatal error.

i. Only Fatal error bit (bit 2) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

ii. Data Link Protocol Error bit (bit 4) in the DUT's Uncorrectable Error Status register is set.

iii. If the error is not masked in the Uncorrectable Error Mask register (bit 4) of AER:

a) If error reporting is enabled in DUT's Device Control register or SERR in Command register is set send ERR_FATAL message.

b) If error reporting is not enabled in DUT's Device Control register and SERR in Command register is not set, *no* ERR_FATAL is sent.

iv. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_FATAL is sent.

v. If all of the above conditions are met, then DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

2) For (Function =1;Function=7;Function++) in DUT:

a. Clear Device Status register and verify that none of the error status bits are set.

b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

c. MACRO_PTC_ PROGRAM (ACK_DLLP_WRONG_SEQ_NUM, ANY_TLP, 1) //ANY_TLP could be Config_Rd_Completion or Unsupported Request depending on whether the Function exists.

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

CASE 1: PCIe1.0.a - Fatal Error - Severity – Non-Fatal - AER implemented

This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is cleared to indicate that this is not a fatal error.

i. Unsupported Request is sent depending on whether the function exists or not.

ii. Only Non-Fatal Error bit (bit 1) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

iii. If the error is not masked in the Uncorrectable Error Mask register (bit 4) of AER:

   a) Data Link Protocol error bit (bit 4) in the Uncorrectable Error Status register of the AER is set.

   b) If error reporting is enabled in DUT's Device Control register, send ERR_NONFATAL message.

   c) If error reporting is not enabled in DUT's Device Control register, *no* ERR_NONFATAL is sent.

iv. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_NONFATAL is sent.

v. If all the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a - Fatal Error - Severity – Fatal - AER implemented

This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is set to indicate that this is a fatal error.

i. Unsupported Request is sent depending on whether the function exists or not.

ii. Only Fatal Error bit (bit 2) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

iii. If the error is not masked in the Uncorrectable Error Mask register (bit 4) of AER:

   a) Data Link Protocol error bit (bit 4) in the Uncorrectable Error Status register of the AER is set.

   b) If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

   c) If error reporting is not enabled in DUT's Device Control register, *no* ERR_FATAL is sent.

iv. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_FATAL is sent.

v. If all of the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 3: PCIe1.0.a - Fatal Error - No AER

This implies DUT has no AER and the Malformed TLP is handled as a fatal error.

i. Unsupported Request is sent depending on whether the function exists or not.

ii. Only Fatal Error bit (bit 2) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

iv. If error reporting is not enabled in DUT's Device Control register, *no* ERR_FATAL is sent.

v. If all of the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 4: PCIe1.1 - Fatal Error - No AER

PCIe2.0 - Fatal Error - No AER

This implies DUT has no AER and the Malformed TLP is handled as a fatal error.

i. Unsupported Request is sent depending on whether the function exists or not.

ii. Only Fatal Error bit (bit 2) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register, send ERR_FATAL message.

iv. If error reporting is not enabled in DUT's Device Control register, *no* ERR_FATAL is sent.

v. If all of the above conditions are met, DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 5: PCIe1.1 – Fatal Error – Severity – Non-Advisory - AER implemented

PCIe2.0 – Fatal Error – Severity – Non-Advisory - AER implemented

This implies DUT implemented AER and, in the Uncorrectable Severity register, Data Link Protocol Error Severity bit (bit 4) is set to indicate that this is a fatal error.

i. Unsupported Request is sent depending on whether the function exists or not.

ii. Only Fatal error bit (bit 2) or Unsupported Request bit (bit 3) in the DUT's Device Status register is set and DUT has not set any other error bits in Device Status register.

iii. Data Link Protocol Error bit (bit 4) in the DUT's Uncorrectable Error Status register is set.

iv. If the error is not masked in the Uncorrectable Error Mask register (bit 4) of AER:

a) If error reporting is enabled in DUT's Device Control register or SERR in Command register is set send ERR_FATAL message.

b) If error reporting is not enabled in DUT's Device Control register and SERR in Command register is not set, *no* ERR_FATAL is sent.

v. If the error is masked in the Uncorrectable Error Mask register (bit 4) of AER, then *no* ERR_FATAL is sent.

vi. If all of the above conditions are met, then DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.3.8.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.3.8.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.4. LCRC and Sequence Number (TLP Receiver)

4.4.1. Test 53-20 BadLCRC ↑

Test Introduction

The intent of this test is to verify that a receiver discards a TLP with bad CRC by sending it a NAK and reporting a BAD TLP error associated with the port.

ASSERTIONS COVERED: DLL.5.3#2

NOTES:

1) Test applies to all PCI Express port types.

2) DATA_BUF – holds the data read back from the device

4.4.1.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC (DUT) is the requester and the PTC is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (CORRUPT_LCRC, CONFIG_RD_COMPLETION, 1)
2) MACRO_PTC_ARM ()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO-PTC_CLEANUP ()
6) Verify that DUT generated a NAK for the CONFIG_RD_COMPLETION TLP. If not, treat it as DUT's failure.
7) Verify that a BAD TLP port error was logged by the DUT for the port connected to the PTC. If not, treat it as DUT's failure.

4.4.1.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and the PTC is the forwarder of that request to the DUT (with bad LCRC).

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_PTC_PROGRAM (CORRUPT_LCRC, CONFIG_RD_REQ, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

i. DUT NAKed the TLP with bad CRC.

ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If the error is not masked in the Correctable Error Mask register (bit 6) of AER:

a) Bad TLP status bit (bit 6) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

iv. If the error is masked in the Correctable Error Mask register (bit 6) of AER, then *no* ERR_COR is sent.

v. If all the above conditions are met, then DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error – No AER implemented

i. DUT NAKed the TLP with bad CRC.

ii. Only Correctable Error bit (bit 0) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If all the above conditions are met, then DUT PASSES the test for Function 0. Otherwise consider it as DUT's failure.

2) For (Function=1;Function=7;Function++) in DUT

a. Clear Device Status register and verify that none of the error status bits are set.

b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)

c. MACRO_PTC_PROGRAM (CORRUPT_LCRC, CONFIG_RD_REQ, 1)

d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.

e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)

f. MACRO-PTC_DISARM ()

g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

CASE 1: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error - AER implemented

i. DUT NAKed the TLP with bad CRC or Unsupported Request was sent depending on whether the function exists or not.

ii. Only Correctable Error bit (bit 0) or Unsupported Request Bit (Bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If the error is not masked in the Correctable Error Mask register (bit 6) of AER:

a) Bad TLP status bit (bit 6) in the Correctable Error Status register of the AER is set.

b) If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

c) If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

vi. If the error is masked in the Correctable Error Mask register (bit 6) of AER, then *no* ERR_COR is sent.

vii. If all the above conditions are met, then DUT PASSES the test. Otherwise consider it as DUT's failure.

CASE 2: PCIe1.0.a, PCIe1.1, PCIe2.0 - Correctable Error – No AER implemented

i. DUT NAKed the TLP with bad CRC or Unsupported Request was sent depending on whether the function exists or not.

ii. Only Correctable Error bit (bit 0) or Unsupported Request Bit (Bit 3) in the DUT's Device Status register is set and DUT did not set any other error bits in Device Status register.

iii. If error reporting is enabled in DUT's Device Control register (bit 0), ERR_COR message is sent by DUT.

iv. If error reporting is not enabled in DUT's Device Control register (bit 0), *no* ERR_COR is sent.

v. If all the above conditions are met, then DUT PASSES the test. Otherwise consider it as DUT's failure.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.4.1.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.4.1.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.4.2. Test 53-31 DuplicateTLP ↑

Test Introduction

The intent of this test is to verify that the duplicate (TLP with the same sequence number associated at the link layer as that in the last 2048 TLPs received) TLPs are handled properly by the DUT.

ASSERTIONS COVERED: DLL.5.3#3.1

NOTES:

1) Test applies to all PCI Express port types.
2) DATA_BUF – holds the data read back from the device

4.4.2.1. RC Test

TOPOLOGY: Platform Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and the PTC is the completer for that request in this test.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.

PROCEDURE:

1) MACRO_PTC_PROGRAM (DUPLICATE_TLP, CONFIG_RD_COMPLETION, 1)
2) MACRO_PTC_ARM()
3) DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_PTC (VENDOR_DEV_ID)
4) MACRO_PTC_STATUS (ACTION COUNT)
5) MACRO_PTC_CLEANUP ()
6) Verify that the RC issues a single coalesced ACK or two ACKs for the duplicate TLPs. If it did, the DUT passes the test.
7) If the DUT did not issue a coalesced ACK or a second ACK as above, log it as DUT's failure.

4.4.2.2. Endpoint Device Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) RC is the CONFIG_RD requester and DUT is the completer for that request.

INITIAL CONDITIONS:

1) Platform is up and running, with drivers for the PTC and the DUT loaded and functioning.
2) PTC is disarmed and no trigger conditions set up.
3) DUT is running default traffic (if any) – that is, no application started yet.

PROCEDURE:

1) For Function 0 in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_ PTC_PROGRAM (DUPLICATE_TLP, CONFIG_RD_REQ, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )
   h. Verify that:
      i. CONFIG_RD_REQ TLP which has been duplicated has received two ACKs or a single coalesced ACK, but a single completion with data from the DUT.
      ii. Verify that the DUT did not set any error bits in Device Status register.
   i. If the DUT meets above criteria, the DUT passes the test for the Function 0.
2) For (Function=1;Function=7;Function++) in DUT:
   a. Clear Device Status register and verify that none of the error status bits are set.
   b. MACRO_PTC_CONFIG_TRACE_BUF(TLP_HEADERS ONLY, UPSTREAM_DIR)
   c. MACRO_ PTC_PROGRAM (DUPLICATE_TLP, CONFIG_RD_REQ, 1)
   d. MACRO_PTC_ARM () // starts the trace buffer capture of all TLP headers from DUT.
   e. DATA_BUF = MACRO_READ_CONFIG_DATA_FROM_DUT (VENDOR_DEVICE_ID)
   f. MACRO-PTC_DISARM ()
   g. MACRO_READ_DATA_FROM_PTC( )

h. Verify that:

i. CONFIG_RD_REQ TLP which has been duplicated has received two ACKs or a single coalesced ACK, but a single completion with data from the DUT or Unsupported Request was sent depending on whether the functions exist or not.

ii. Verify that the DUT did not set any error bits in Device Status register other than Unsupported Request Detected (bit 3) depending on whether the functions exist or not.

i. If the DUT meets above criteria, the DUT passes the test.

3) If the DUT fails for any Function, treat the overall result as DUT's failure.

4.4.2.3. Switch Downstream Port Test

TOPOLOGY: Switch Test Topology, PTC in Platform Test mode

SECTION NOTES:

1) Algorithm same as in the RC test case except the DUT is a Switch's downstream port.

4.4.2.4. Switch Upstream Port / Bridge Test

TOPOLOGY: Endpoint Test Topology, PTC in Add-in card Test mode

SECTION NOTES:

1) Algorithm same as in the Endpoint device test case except the DUT is the Switch's upstream port.

4.5. Reserved Bits in Training Sequences

4.5.1. PCIe 2.0 Endpoint Device Test

Initial conditions: Platform (Test System, which is PCIe 2.0, 5.0 GT/s data rate capable) and the DUT are powered, platform initiates fundamental reset to the DUT, and platform is up and running, with drivers for the test platform loaded and functioning. The link in Detect state.

Note: Once all criteria for the platform LTSSM to move to the next LTSSM state have been met (as specified in Chapter 4 of the PCI Express Base Specification 2.0), the platform must make the transition to the next state within 500 μs.

1) Platform initiates link training into L0 state, advertising 2.5 GT/s data rate support. For this training all the bits in the TS1/TS2 that are reserved in the PCIe 2.0 Base Specification are set to 0 in every TS1/TS2 transmitted. Verify that link training is successful by checking that Idle Symbols are transmitted by the DUT at 2.5 GT/s, or by checking that the link is operational at 2.5 GT/s upon entering L0. To check that the link is operational, platform will send InitFC DLLPs to complete flow control initialization, send a config read to register 0 (vendor / device ID), and verify that valid data is returned by the DUT. Otherwise, log as DUT failure and note LTSSM state at which link training is stuck.
2) Platform brings link into Detect state by initiating fundamental reset to the DUT.
3) Platform initiates link training into L0 state advertising 2.5 GT/s data rate support. For this training all the bits in the TS1/TS2 that are reserved in the PCIe 2.0 Base Specification are set to 1 in every TS1/TS2 transmitted:
   a. Bit 0 in Byte 4 (Data Rate Identifier)
   b. Bits 3:5 in Byte 4 (Data Rate Identifier)
   c. Bits 5:7 in Byte 5 (Training Control)
   d. Bit 4 in Byte 5 of TS2 (Training Control)

   Verify that link training is successful by checking that Idle Symbols are transmitted by the DUT at 2.5 GT/s, or by checking that the link is operational at 2.5 GT/s upon entering L0. To check that the link is operational, platform will send InitFC DLLPs to complete flow control initialization, send a config read to register 0 (vendor / device ID), and verify that valid data is returned by the DUT. Otherwise, log as DUT failure and note LTSSM state at which link training is stuck.
4) Steps 2 and 3 are repeated; this time platform initiates link retraining by bringing the link to Recovery.RcvrLock, to Recovery.RcvrCfg, to Recovery.Idle, and back to L0 state. During Recovery state the specified reserved bits in step 3 are still set to 1 for every TS1/TS2 transmitted. Verify that link retraining is successful by checking that Idle Symbols are transmitted by the DUT at 2.5 GT/s, or by checking that the link is operational at 2.5 GT/s upon re-entering L0. To check that the link is operational, platform will send a config read to register 0 (vendor / device ID), and verify that valid data is returned by the DUT. Otherwise, log as DUT failure and note LTSSM state at which link training is stuck.

5) Platform brings the link back to Detect state by initiating fundamental reset to the DUT. Steps 1-4 are repeated, with the platform advertising 5.0 GT/s support. For a 5.0 GT/s capable DUT, at every step, the platform must verify that the link training is successful at 5.0 GT/s.

In case of failure an optional detailed version of the test may be executed in which each reserved bit mentioned in step 3 above is asserted individually for each TS1/TS2 in the session and correct link training from Detect into L0 and link retraining from Recovery into L0 is verified under these conditions.

4.5.2. PCIe 2.0 RC Test

The algorithm is the same as a PCIe 2.0 Endpoint except the DUT is a downstream port on a system and the test device is in Endpoint emulation mode. In endpoint emulation mode, the fundamental reset is replaced by a power cycle of the system. In endpoint emulation mode, the platform can only verify that the link is in L0 state at the correct link speed by checking that Idle Symbols are transmitted by the DUT at the correct link speed.

4.5.3. PCIe 2.0 Switch and Bridge Test

For the upstream port of a switch or bridge, the PCIe 2.0 Endpoint test is run.
For the downstream port of a switch or bridge, the PCIe 2.0 RC test is run.

A

A. MACROS

A.1. Protocol Test Card (PTC) Related

These Macros shall apply to any test equipment that has the test capabilities listed previously. As noted earlier, in this document such capable test equipment is referred to as the PTC. The implementation details of the Macros are test equipment specific.

A.1.1. MACRO_POLL_PTC_FOR_LINK_RETRAINING ()

Description: Monitor the link status after a link retrain. This Macro will periodically check the link status for retraining to complete. If the link does not successfully retrain within the specified period, this will be treated as test failure and test aborted.

A.1.2. MACRO_PTC_ARM ()

Description: Enables the PTC card to act on the test conditions already programmed into it. This along with the DISARM capability gives the software control on when the PTC shall apply the test conditions on the DUT. This will also enable capturing the traffic flowing through the PTC into its trace buffer.

A.1.3. MACRO_PTC_DISARM ()

Description: Disables the PTC card from applying the test conditions to any traffic flowing through and this will also stop trace buffer acquisition.

A.1.4. MACRO_PTC_STATUS (ACTION COUNT)

Description: Determine how far the PTC is in executing the test command set up. This Macro will periodically poll the action count for a maximum of 10 ms for the count to go to zero. If the count does not reach zero in that period, this will be treated as test failure and test aborted.

ActionCount – 0 => PTC executed on the test command

!= 0 > PTC is still looking to assert the test command

A.1.5. MACRO_PTC_PROGRAM (PTC_ACTION, PATTERN_TO_MATCH, ACTION COUNT)

Description: Program the PTC to generate desired test condition.

PTC_ACTION – Refers to the test condition the PTC will generate

PATTER_TO_MATCH – A matching condition (TLP or DLLP as appropriate) on which the PTC will generate the test condition.

ACTION COUNT – A non zero value that specifies how many time the PTC will generate the test condition

A.1.6. MACRO _READ_CONFIG_DATA_FROM_PTC (QUAL)

Description: Read configuration data from the PTC. Amount of data read depends upon the qualifier

QUAL – VENDOR_DEV_ID – *default* – first DWORD

- PCI_COMPATIBLE- first 256 bytes or as many as specified
  - Any other specific fields in PCI compatible space
  - Extended Config space
  - Any specific fields in extended Config space

A.1.7. MACRO_ READ_CONFIG_DATA_FROM_KEP (QUAL)

Description: Read configuration data from the PTC. This could be a byte, word, dword, or greater depending upon the qualifier.

QUAL – VENDOR_DEV_ID – *default* – first DWORD

- PCI_COMPATIBLE- first 256 bytes or as many as specified
  - Any other specific fields in PCI compatible space
  - Extended Config space
  - Any specific fields in extended Config space

A.1.8. MACRO_ READ_CONFIG_DATA_FROM_DUT (QUAL)

Description: Read configuration data from the DUT. This could be a byte, word, dword, or greater depending upon the qualifier.

QUAL – VENDOR_DEV_ID – *default* – first DWORD

- PCI_COMPATIBLE- first 256 bytes or as many specified
  - Any other specific fields in PCI compatible space
  - Extended Config space
  - Any specific fields in extended Config space

A.1.9. MACRO_READ_DATA_FROM_PTC (START_ADDR, BYTE_COUNT, BAR_NUM)

Description: Read from the memory behind the PTC's BAR. This will be the trace buffer memory.

START_ADDR – Start address at which the read begins

BYTE_COUNT – Number of bytes to read

BAR_NUM – In case there is more than one BAR implemented by the test equipment, this will specify the BAR behind which the memory read shall take place.

A.1.10. MACRO_READ_DATA_FROM_KEP (START_ADDR, BYTE_COUNT, BAR_NUM)

Description: Read from the memory behind the KEP's BAR

START_ADDR – Start address at which the read begins

BYTE_COUNT – Number of bytes to read

BAR_NUM – In case there is more than one BAR implemented by the test equipment, this will specify the BAR behind which the memory read shall take place.

A.1.11. MACRO_WRITE_DATA_TO_PTC (START_ADDR, DATA_PATTERN, BYTE_COUNT, BAR_NUM)

Description: Write to the memory behind the PTC' BAR. There is at least one BAR in the PTC.

START_ADDR – Start address at which the write begins

DATA – A DWORD pattern that will be repeated up to the byte count. In the case of partial fills, the lower bytes of the pattern shall be used as needed.

BYTE_COUNT – Number of bytes to write

BAR_NUM – In case there is more than one BAR implemented by the test equipment, this will specify the BAR behind which the memory write shall take place.

A.1.12. MACRO_WRITE_DATA_TO_KEP (START_ADDR, DATA_PATTERN, BYTE_COUNT, BAR_NUM)

Description: Write to the memory behind the KEP's BAR. There is at least one BAR present in the KEP.

START_ADDR – Start address at which the write begins

DATA – A DWORD pattern that will be repeated up to the byte count. In the case of partial fills, the lower bytes of the pattern shall be used as needed.

BYTE_COUNT – Number of bytes to write

BAR_NUM – In case there is more than one BAR implemented by the test equipment, this will specify the BAR behind which the memory write shall take place.

A.1.13. MACRO_PTC_CLEANUP ()

Description: Disarm the PTC and do any other clean up needed on the PTC (test equipment).

A.2. GENERAL

A.2.1. MACRO_ENABLE_LINK (ACTION)

Description: Enable and disable the link at any of the downstream ports on the PTC device.

ACTION – YES or NO

A.2.2. MACRO_WRITE_CONFIG_DATA_TO_DUT(ACTION)

Description: Enable or disable the transmission of error messages by the DUT.

ACTION – MSG_MSG_OFF: Disable error messaging

ERR_MSG_ON – Enable error messaging

5. Acknowledgements

Shiva Aditham	Intel Corporation
Will Atherton	IBM
Gord Caruk	Advanced Micro Devices, Inc.
Rick Eads	Agilent Technologies, Inc.
Michael Engbretson	Tektronix, Inc.
Dan Froelich	Intel Corporation
Gordon Getty	Agilent Technologies, Inc.

Bent Hessen-Schmidt	Bertscope
Betty Luk	Advanced Micro Devices, Inc.
Steve Manning	Advanced Micro Devices, Inc.
Dan Neal	QuestTech
Michael Pasumansky	LeCroy
Bill Simms	Nvidia
Richard Solomon	LSI Logic Corporation
John Wiedemeier	LeCroy